Lab 9-10 – Nanoprocessor Design Competition

CS1050 Computer Organization and Digital Design

- **Group Members : Muthuwana M.A.N.R.**
  Nandaka T.H.T.D.
- **Index : 210400N**
  210406M

1) The assigned lab task in few sentences

- ✓ The assigned lab task involves designing a 4-bit nanoprocessor capable of executing a set of instructions. The specific requirements for the lab task are as follows:
- ✓ Design and develop a 4-bit add/subtract unit: This unit should be capable of adding and subtracting numbers represented using 2's complement. You can modify the 4-bit Ripple Carry Adder (RCA) from Lab 3 to implement this component.
- ✓ Design and develop a 3-bit adder: This unit is used to increment the Program Counter (PC).
- ✓ Design and develop a 3-bit Program Counter (PC): The program counter needs to be reset to 0 when required. Build it using D Flip Flops with a clear/reset input.
- ✓ Build k-way b-bit multiplexers: A k-way b-bit multiplexer can take in k inputs, each with b bits, and select one of the k groups of b bits based on control bits. Build a 2-way 3-bit multiplexer, a 2-way 4-bit multiplexer, and an 8-way 4-bit multiplexer.
- ✓ Build the Register Bank: The Register Bank should contain 8, 4-bit registers (R0 to R7). Initialize the value of R0 to all 0s .
- ✓ Build the Program ROM: This component stores the assembly program by using lookup tables. .
- ✓ Design the Instruction Decoder: The Instruction Decoder circuit should activate necessary components based on the instructions to be executed. Design the internal logic to properly execute the instructions provided in Table 1.
- ✓ Write an Assembly program: Write an Assembly program that calculates the total of all integers between 1 and 3. Use only the instructions specified in Table 1, and store the final answer in Register R7. Convert the Assembly program to machine code and hard code it into the ROM.
- ✓ Connect inputs and outputs: Connect the output of R7 to a set of LEDs and a 7-segment display. Connect LD14 and LD15 for zero and carry flags, respectively.

- ✓ Test the nanoprocessor on the BASYS 3 development board: Verify the functionality of your nanoprocessor by testing it on the hardware. Demonstrate the circuit to the instructor and explain how your design works. Each team member's contribution to the project should be clearly described.

- ✓ Remember to refer to the provided lab materials, such as circuit diagrams, instruction set, and guidelines, for detailed instructions and specifications.

2) Assembly program and its machine code representation

MACHINE CODE	***ASSEMBLY CODE***
"101110000001"	***MOVI R7,1***
"101100000010"	***MOVI R6,2***
"101010000011"	***MOVI R5,3***
"001111100000"	***ADD R7 <- R7 + R6***
"001111010000"	***ADD R7 <- R7 + R5***
"110000000000"	***JZR R0,0***
"000000000000"	
"000000000000"	

3) All VHDL codes

- *NanoProcessor_VHDL*

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 07.06.2023 15:13:15
-- Design Name:
-- Module Name: NanoProcessor - Behavioral
-- Project Name:
```

```
-- Target Devices:
-- Tool Versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------------


library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--use IEEE.NUMERIC_STD.ALL;

-- Uncomment the following library declaration if instantiating
-- any Xilinx leaf cells in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

entity NanoProcessor is
    Port (
          pushButton : in STD_LOGIC;
```

```
        jump_Flag: out std_logic;
        Address_to_jump:out std_logic_vector(2 downto 0);

        Register_check_for_jump:out std_logic_vector(3 downto 0);
        LED_R0:out std_logic_vector(3 downto 0);

        Imm_Value:out std_logic_vector(3 downto 0);
        Clk : in STD_LOGIC;
        nextInsVal : out std_logic_vector(2 downto 0);
        RegisterEnable:out std_logic_vector(2 downto 0);
        RegisterBank_DataIn: out std_logic_vector(3 downto 0);
        LED : out STD_LOGIC_VECTOR (3 downto 0);
        LED_R6 : out std_logic_vector(3 downto 0);
        LED_R5 : out std_logic_vector(3 downto 0);
        LED_OVERFLOW:out STD_LOGIC;
        LED_ZERO:out STD_LOGIC;
        Instruction : out std_logic_vector(11 downto 0));
end NanoProcessor;

architecture Behavioral of NanoProcessor is
COMPONENT Slow_Clk is
   Port ( Clk_in : in STD_LOGIC;
        Clk_out : out STD_LOGIC);
end COMPONENT;
COMPONENT Mux_8_Way_4_Bit is  --MUX 8 X 1 (4 BIT)
   Port ( Reg_Sel : in STD_LOGIC_VECTOR (2 downto 0);
        R0 : in STD_LOGIC_VECTOR (3 downto 0);
        R1 : in STD_LOGIC_VECTOR (3 downto 0);
```

```
        R2 : in STD_LOGIC_VECTOR (3 downto 0);
        R3 : in STD_LOGIC_VECTOR (3 downto 0);
        R4 : in STD_LOGIC_VECTOR (3 downto 0);
        R5 : in STD_LOGIC_VECTOR (3 downto 0);
        R6 : in STD_LOGIC_VECTOR (3 downto 0);
        R7 : in STD_LOGIC_VECTOR (3 downto 0);
        Mux_Out  : out STD_LOGIC_VECTOR (3 downto 0));
end COMPONENT;
COMPONENT Mux_2_Way_4_Bit is --MUX 2 X 1 (4 BIT)
   Port ( ImmVal : in STD_LOGIC_VECTOR (3 downto 0);
        ASU : in STD_LOGIC_VECTOR (3 downto 0);
        out_RegIn : out STD_LOGIC_VECTOR (3 downto 0);
        LS : in STD_LOGIC);
end COMPONENT;
COMPONENT Way_2_Bit3_Mux is  --MUX 2 X 1 (3 BIT)
   Port ( Jump_Address : in STD_LOGIC_VECTOR (2 downto 0);
        Adder_3_Out: in STD_LOGIC_VECTOR (2 downto 0);
        Sel_Jump : in STD_LOGIC;
        Output : out STD_LOGIC_VECTOR (2 downto 0));
end COMPONENT;
COMPONENT AddSubUnit IS
   Port ( A0 : in STD_LOGIC;
        A1 : in STD_LOGIC;
        A2 : in STD_LOGIC;
        A3 : in STD_LOGIC;
        M : in STD_LOGIC;
        B0 : in STD_LOGIC;
        B1 : in STD_LOGIC;
```

```
        B2 : in STD_LOGIC;
        B3 : in STD_LOGIC;
        S0 : out STD_LOGIC;
        S1 : out STD_LOGIC;
        S2 : out STD_LOGIC;
        S3 : out STD_LOGIC;
        C_OUT : out STD_LOGIC;
        OVERFLOW: out STD_LOGIC;
        ZERO: out STD_LOGIC );
end COMPONENT;
COMPONENT RCA_4 is
   Port ( A : in STD_LOGIC_VECTOR(2 DOWNTO 0);
        B : in STD_LOGIC_VECTOR(2 DOWNTO 0);
        C_in : in STD_LOGIC;
        S : out STD_LOGIC_VECTOR(2 DOWNTO 0);
        C_out : out STD_LOGIC);
END COMPONENT ;
COMPONENT ProgramROM is  --ROM
   Port ( address : in STD_LOGIC_VECTOR (2 downto 0);
        data : out STD_LOGIC_VECTOR (11 downto 0));
end COMPONENT;
COMPONENT InstructionDecoder is --Instruction Decoder
   Port ( InsBus : in STD_LOGIC_VECTOR (11 downto 0);  --instruction
bus
        RCJ : in STD_LOGIC_VECTOR (3 downto 0);      --Register check
for jump
        RegEn : out STD_LOGIC_VECTOR (2 downto 0);   --Register Enable
        LS : out STD_LOGIC;                    --Load Select
```

```
        ImVal : out STD_LOGIC_VECTOR (3 downto 0);   --Immediate
Value
        RegS1 : out STD_LOGIC_VECTOR (2 downto 0);   --Register Select 1
        RegS2 : out STD_LOGIC_VECTOR (2 downto 0);   --Register Select 2
        AddSubS : out STD_LOGIC;                     --Add/Sub Select
        JumpFlag : out STD_LOGIC;                    --JumpFlag
        ATJ : out STD_LOGIC_VECTOR (2 downto 0));    --Address to jump
    end COMPONENT;
    COMPONENT Reg_Bank is
      Port ( Clk : in STD_LOGIC;
        Reg_EN : in STD_LOGIC_VECTOR (2 downto 0);
        push_button :in std_logic;
        Reg_Bank_In : in STD_LOGIC_VECTOR (3 downto 0);
        Reg_B_Out_0 : out STD_LOGIC_VECTOR (3 downto 0);
        Reg_B_Out_1 : out STD_LOGIC_VECTOR (3 downto 0);
        Reg_B_Out_2 : out STD_LOGIC_VECTOR (3 downto 0);
        Reg_B_Out_3 : out STD_LOGIC_VECTOR (3 downto 0);
        Reg_B_Out_4 : out STD_LOGIC_VECTOR (3 downto 0);--register
banks
        Reg_B_Out_5 : out STD_LOGIC_VECTOR (3 downto 0);
        Reg_B_Out_6 : out STD_LOGIC_VECTOR (3 downto 0);
        Reg_B_Out_7 : out STD_LOGIC_VECTOR (3 downto 0)
        );
    end COMPONENT;
    COMPONENT ProgramCounter_3Bit is
      Port ( Reset_PushButton : in STD_LOGIC;
        Clk : in STD_LOGIC;
        pc_in : in STD_LOGIC_VECTOR (2 downto 0);
        MemSel : out STD_LOGIC_VECTOR (2 downto 0));
```

```
end COMPONENT;

SIGNAL CLOCK:STD_LOGIC;

-----MUX 8 X 1

SIGNAL MUX_0_OUT,MUX_1_OUT:STD_LOGIC_VECTOR(3
DOWNTO 0);

SIGNAL
R0,R1,R2,R3,R4,R5,R6,R7,REG_BANK_IN:STD_LOGIC_VECTOR(3
DOWNTO 0); --MUX 8X1 AND REG OUTS CONNECTERS

SIGNAL REG_S1,REG_S2,REG_EN:STD_LOGIC_VECTOR(2 DOWNTO
0);


SIGNAL LS_s:STD_LOGIC;

SIGNAL IMMVAL,ASU_s:STD_LOGIC_VECTOR(3 DOWNTO 0);

-----ADD SUB UNIT

SIGNAL M,OVERFLOW,ZERO,C_OUT:STD_LOGIC;

------DECODER

SIGNAL InsBus: STD_LOGIC_VECTOR(11 DOWNTO 0);

SIGNAL JUMPFLAG:STD_LOGIC;

SIGNAL ATJ:STD_LOGIC_VECTOR(2 DOWNTO 0);

---ROM

SIGNAL MEMORYSELECT:STD_LOGIC_VECTOR(2 DOWNTO 0);

--3 BIT ADDER

SIGNAL THREE_BIT_ADDER_OUT:STD_LOGIC_VECTOR(2
DOWNTO 0);

SIGNAL C_OUTT:STD_LOGIC;

---PC

SIGNAL NextIns:STD_LOGIC_VECTOR(2 DOWNTO 0);


SIGNAL SLOW_CLOCK_OUT:STD_LOGIC;
```

```
signal pushButton_s : std_logic;
-------------------------------------------------------------------------------------------
-
begin
   pushButton_s <= pushButton;
REGBANK :Reg_Bank
   port map(
   Clk=>CLOCK,
   Reg_EN=>REG_EN,
   Reg_Bank_In=>REG_BANK_IN,
   Reg_B_Out_0=>R0,
   Reg_B_Out_1=>R1,
   Reg_B_Out_2=>R2,
   Reg_B_Out_3=>R3,
   Reg_B_Out_4=>R4,
   Reg_B_Out_5=>R5,
   Reg_B_Out_6=>R6,
   Reg_B_Out_7=>R7,
   push_button=>pushButton_s
   );
MUX8X1_0:Mux_8_Way_4_Bit
   port map(
    Reg_Sel =>REG_S1,
    R0 =>R0,
    R1 =>R1,
    R2 =>R2,
    R3 =>R3,
    R4 =>R4,
    R5 =>R5,
```

```
    R6 =>R6,
    R7 =>R7,
    Mux_Out =>MUX_0_OUT);
MUX8X1_1:Mux_8_Way_4_Bit
    port map(
    Reg_Sel =>REG_S2,
    R0 =>R0,
    R1 =>R1,
    R2 =>R2,
    R3 =>R3,
    R4 =>R4,
    R5 =>R5,
    R6 =>R6,
    R7 =>R7,
    Mux_Out =>MUX_1_OUT);
MUX2X1_4BIT: Mux_2_Way_4_Bit
   port map(
   out_RegIn=>REG_BANK_IN,
   ImmVal=>IMMVAL,
   ASU=>ASU_s,
   LS=>LS_s);
MUX2X1_3BIT:Way_2_Bit3_Mux
   port map(
   Jump_Address=>ATJ,
   Adder_3_Out=>THREE_BIT_ADDER_OUT,
   Sel_Jump=>JUMPFLAG,
   Output=>NextIns
   );
```

```
ADDSUBUNIT_0:  AddSubUnit--ADD SUB UNIT
   Port map (
   A0 =>MUX_0_OUT(0),
   A1 =>MUX_0_OUT(1),
   A2 =>MUX_0_OUT(2),
   A3 =>MUX_0_OUT(3),
   M  =>M,
   B0 =>MUX_1_OUT(0),
   B1 =>MUX_1_OUT(1),
   B2 =>MUX_1_OUT(2),
   B3 =>MUX_1_OUT(3),
   S0 =>ASU_s(0),
   S1 =>ASU_s(1),
   S2 =>ASU_s(2),
   S3 =>ASU_s(3),
   C_OUT =>C_OUT,
   ZERO =>ZERO,
   OVERFLOW => OVERFLOW);
INSTRUCTIONDECODER0:InstructionDecoder
   Port map(
   InsBus=>InsBus,
   RCJ(0)=>MUX_0_OUT(0),
   RCJ(1)=>MUX_0_OUT(1),
   RCJ(2)=>MUX_0_OUT(2),
   RCJ(3) =>MUX_0_OUT(3),
   RegEn =>REG_EN,
   LS =>LS_s,
   ImVal=>IMMVAL,
```

```
    RegS1=>REG_S1,
    RegS2 =>REG_S2,
    AddSubS =>M,
    JumpFlag=>JUMPFLAG,
    ATJ=>ATJ);
THREEBITADDER:RCA_4
    Port map(
    A=>MEMORYSELECT,
    C_in=>'0',
    B => "001",
    S=>THREE_BIT_ADDER_OUT,
    C_out=>C_OUTT
    );
ROM :ProgramROM
    Port map(
    address=>MEMORYSELECT,
    data=>InsBus);
SLOWCLOCK :Slow_Clk
    Port map(
    Clk_in=>Clk,
    Clk_out=>SLOW_CLOCK_OUT
    );
PROGRAMCOUNTER:ProgramCounter_3Bit
    Port map(
    Reset_PushButton=>pushButton_s,
    --Clk=>SLOW_CLOCK_OUT,
    Clk=>CLOCK,
    pc_in=>NextIns,
```

```
    MemSel=>MEMORYSELECT
    );
    --SET OUTPUTS TO LEDS---------------------------------

    CLOCK <= Clk;
    nextInsVal <= NextIns;
--   pushButton_s <= pushButton;
    LED<=R7;
    LED_R6 <= R6;
    LED_R5 <= R5;
    LED_OVERFLOW<=OVERFLOW;
    LED_ZERO<=ZERO;
    RegisterEnable<=REG_EN;
    RegisterBank_DataIn<=REG_BANK_IN;
    Imm_Value<=IMMVAL;
    Instruction <= InsBus;
    jump_Flag<=JUMPFLAG;
    Address_to_jump<=ATJ;
    Register_check_for_jump<=MUX_0_OUT;
    LED_R0<=R0;
end Behavioral;
```

- *Register Bank VHDL*

```
----------------------------------------------------------------------------
-- Company:
-- Engineer:
```

```
--
-- Create Date: 05/26/2023 07:28:23 PM
-- Design Name:
-- Module Name: Reg_Bank - Behavioral
-- Project Name:
-- Target Devices:
-- Tool Versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------------


library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--use IEEE.NUMERIC_STD.ALL;

-- Uncomment the following library declaration if instantiating
-- any Xilinx leaf cells in this code.
--library UNISIM;
```

```
--use UNISIM.VComponents.all;


entity Reg_Bank is
   Port ( Clk : in STD_LOGIC;
          Reg_EN : in STD_LOGIC_VECTOR (2 downto 0);
          Reg_Bank_In : in STD_LOGIC_VECTOR (3 downto 0);
          push_button:in std_logic;  ---push_button=1 =>reset
          Reg_B_Out_0 : out STD_LOGIC_VECTOR (3 downto 0);
          Reg_B_Out_1 : out STD_LOGIC_VECTOR (3 downto 0);
          Reg_B_Out_2 : out STD_LOGIC_VECTOR (3 downto 0);
          Reg_B_Out_3 : out STD_LOGIC_VECTOR (3 downto 0);
          Reg_B_Out_4 : out STD_LOGIC_VECTOR (3 downto 0);--register
banks
          Reg_B_Out_5 : out STD_LOGIC_VECTOR (3 downto 0);
          Reg_B_Out_6 : out STD_LOGIC_VECTOR (3 downto 0);
          Reg_B_Out_7 : out STD_LOGIC_VECTOR (3 downto 0)
          );
end Reg_Bank;

architecture Behavioral of Reg_Bank is
component Reg
   port(
      D : in STD_LOGIC_VECTOR (3 downto 0);
      En : in STD_LOGIC;
      Clk : in STD_LOGIC;
      RESET: IN STD_LOGIC;
      Q : out STD_LOGIC_VECTOR (3 downto 0)
    );
 end component;
```

```
component Decoder_3_to_8
      Port (
      I : in STD_LOGIC_VECTOR (2 downto 0);
      EN : in STD_LOGIC;
      Y : out STD_LOGIC_VECTOR (7 downto 0)
      );
end component;

signal decoder_out :  std_LOGIC_VECTOR(7 downto 0);
-- signal
regout0,regout1,regout2,regout3,regout4,regout5,regout6,regout7:std_logic_
vector(3 downto 0);

begin
--------------------------------------------------------------------------
   Decoder_reg_enable : Decoder_3_to_8
      port map(
         I => Reg_EN,
         EN => '1',
         Y => decoder_out
         );
--------------------------------------------------------------------------

        register_0 : Reg
           port map(
              D => "0000",
              EN=> decoder_out(0),
              Clk => Clk,
```

```
      RESET=>push_button,
      Q=> Reg_B_Out_0
  );
register_1 : Reg
   port map(
      D => Reg_Bank_In,
      EN=> decoder_out(1),
      Clk => Clk,
      RESET=>push_button,
      Q=> Reg_B_Out_1
 );
 register_2 : Reg
   port map(
      D => Reg_Bank_In,
      EN=> decoder_out(2),
      Clk => Clk,
      RESET=>push_button,
      Q=> Reg_B_Out_2
  );
register_3 : Reg
    port map(
      D => Reg_Bank_In,
      EN=> decoder_out(3),
      Clk => Clk,
      RESET=>push_button,
      Q=> Reg_B_Out_3
  );
 register_4 : Reg
```

```
    port map(
      D => Reg_Bank_In,
      EN=> decoder_out(4),
      Clk => Clk,
      RESET=>push_button,
      Q=> Reg_B_Out_4
 );
register_5 : Reg
     port map(
      D => Reg_Bank_In,
      EN=> decoder_out(5),
      Clk => Clk,
      RESET=>push_button,
      Q=> Reg_B_Out_5
 );
register_6 : Reg
     port map(
      D => Reg_Bank_In,
      EN=> decoder_out(6),
      Clk => Clk,
      RESET=>push_button,
      Q=> Reg_B_Out_6
                        );
register_7 : Reg
     port map(
      D =>Reg_Bank_In,
      EN=> decoder_out(7),
      Clk => Clk,
```

```
            RESET=>push_button,
            Q=> Reg_B_Out_7
        );

    end Behavioral;
```

- *8 Way 4 Bit Multiplexer*

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 06/01/2023 05:04:59 PM
-- Design Name:
-- Module Name: Mux_8_Way_4_Bit - Behavioral
-- Project Name:
-- Target Devices:
-- Tool Versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------------


library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--use IEEE.NUMERIC_STD.ALL;

-- Uncomment the following library declaration if instantiating
-- any Xilinx leaf cells in this code.
```

```
--library UNISIM;
--use UNISIM.VComponents.all;

entity Mux_8_Way_4_Bit is
   Port ( Reg_Sel : in STD_LOGIC_VECTOR (2 downto 0);
          R0 : in STD_LOGIC_VECTOR (3 downto 0);
          R1 : in STD_LOGIC_VECTOR (3 downto 0);
          R2 : in STD_LOGIC_VECTOR (3 downto 0);
          R3 : in STD_LOGIC_VECTOR (3 downto 0);
          R4 : in STD_LOGIC_VECTOR (3 downto 0);
          R5 : in STD_LOGIC_VECTOR (3 downto 0);
          R6 : in STD_LOGIC_VECTOR (3 downto 0);
          R7 : in STD_LOGIC_VECTOR (3 downto 0);
          Mux_Out  : out STD_LOGIC_VECTOR (3 downto 0));
end Mux_8_Way_4_Bit;

architecture Behavioral of Mux_8_Way_4_Bit is
component Decoder_3_to_8
   port(I : in STD_LOGIC_VECTOR (2 downto 0);
   EN : in STD_LOGIC;
   Y : out STD_LOGIC_VECTOR (7 downto 0));
end component;

signal Dec_Out : std_logic_vector(7 downto 0);
signal Mux_Output : std_logic_vector(3 downto 0);
signal D0 : std_logic_vector(3 downto 0);
signal A0 : std_logic_vector(3 downto 0);
signal D1 : std_logic_vector(3 downto 0);
signal A1 : std_logic_vector(3 downto 0);
signal D2 : std_logic_vector(3 downto 0);
signal A2 : std_logic_vector(3 downto 0);
signal D3 : std_logic_vector(3 downto 0);
signal A3 : std_logic_vector(3 downto 0);
signal D4 : std_logic_vector(3 downto 0);
signal A4 : std_logic_vector(3 downto 0);
signal D5 : std_logic_vector(3 downto 0);
signal A5 : std_logic_vector(3 downto 0);
signal D6 : std_logic_vector(3 downto 0);
signal A6 : std_logic_vector(3 downto 0);
signal D7 : std_logic_vector(3 downto 0);
signal A7 : std_logic_vector(3 downto 0);


begin
Decoder_3_to_8_0 : Decoder_3_to_8
   port map (
```

```
        I => Reg_Sel,
        EN => '1',
        Y => Dec_Out
    );
    A0 <= (others=> Dec_Out(0));
    A1 <= (others=> Dec_Out(1));
    A2 <= (others=> Dec_Out(2));
    A3 <= (others=> Dec_Out(3));
    A4 <= (others=> Dec_Out(4));
    A5 <= (others=> Dec_Out(5));
    A6 <= (others=> Dec_Out(6));
    A7 <= (others=> Dec_Out(7));

    D0 <= R0 AND A0;
    D1 <= R1 AND A1;
    D2 <= R2 AND A2;
    D3 <= R3 AND A3;
    D4 <= R4 AND A4;
    D5 <= R5 AND A5;
    D6 <= R6 AND A6;
    D7 <= R7 AND A7;

    Mux_Out <= D0 or D1 or D2 or D3 or D4 or D5 or D6 or D7;

end Behavioral;
```

- *Multiplexer 2 way 4 Bit*

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 05.06.2023 12:10:21
-- Design Name:
-- Module Name: Mux_2_Way_4_Bit - Behavioral
-- Project Name:
-- Target Devices:
-- Tool Versions:
-- Description:
--
```

```
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------------


library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--use IEEE.NUMERIC_STD.ALL;

-- Uncomment the following library declaration if instantiating
-- any Xilinx leaf cells in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

entity Mux_2_Way_4_Bit is
    Port ( ImmVal : in STD_LOGIC_VECTOR (3 downto 0);
           ASU : in STD_LOGIC_VECTOR (3 downto 0);
           out_RegIn : out STD_LOGIC_VECTOR (3 downto 0);
           LS : in STD_LOGIC);
end Mux_2_Way_4_Bit;

architecture Behavioral of Mux_2_Way_4_Bit is

signal ls_s : std_logic;
signal asu_s : std_logic_vector(3 downto 0);
signal immVal_s : std_logic_vector(3 downto 0);
signal output : std_logic_vector(3 downto 0);
begin

    ls_s <= LS;
    asu_s <= ASU;
    immVal_s <= ImmVal;
    process (ls_s, asu_s, immVal_s) begin
        if ls_s='1' then
            output<=asu_s;
        else
            output<=immVal_s;
        end if;
    end process;
```

```
   out_RegIn <= output;
end Behavioral;
```

- *Add Sub Unit*

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 05.06.2023 11:31:04
-- Design Name:
-- Module Name: AddSubUnit - Behavioral
-- Project Name:
-- Target Devices:
-- Tool Versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--use IEEE.NUMERIC_STD.ALL;

-- Uncomment the following library declaration if instantiating
-- any Xilinx leaf cells in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

entity AddSubUnit is
   Port ( A0 : in STD_LOGIC;
          A1 : in STD_LOGIC;
          A2 : in STD_LOGIC;
          A3 : in STD_LOGIC;
          M : in STD_LOGIC;
          B0 : in STD_LOGIC;
```

```
        B1 : in STD_LOGIC;
        B2 : in STD_LOGIC;
        B3 : in STD_LOGIC;
        S0 : out STD_LOGIC;
        S1 : out STD_LOGIC;
        S2 : out STD_LOGIC;
        S3 : out STD_LOGIC;
        C_OUT : out STD_LOGIC;
        OVERFLOW: out STD_LOGIC;
        ZERO: out STD_LOGIC );
end AddSubUnit;

architecture Behavioral of AddSubUnit is
   component FA
      port (
      A: in std_logic;
      B: in std_logic;
      C_in: in std_logic;
      S: out std_logic;
      C_out: out std_logic);
   end component;
   SIGNAL FA0_C, FA1_C, FA2_C, FA3_C ,V,CARRY: std_logic;
   SIGNAL ONES0,ONES1,ONES2,ONES3 :std_logic; --For ones complement
   SIGNAL S_0,S_1,S_2,S_3 :STD_LOGIC;
begin
   ONES0<=B0 XOR M;
   FA_0 : FA
      port map (
          A => A0,
          B => ONES0,
          C_in => M , --M
          S => S_0,
          C_out => FA0_C);

   ONES1<=B1 XOR M;
   FA_1 : FA
      port map (
          A => A1,
          B => ONES1,
          C_in => FA0_C,
          S => S_1,
          C_out => FA1_C);

    ONES2<=B2 XOR M;
    FA_2 : FA
       port map (
```

```
            A => A2,
            B => ONES2,
            C_in => FA1_C,
            S => S_2,
            C_out => FA2_C);

    ONES3<=B3 XOR M;
    FA_3 : FA
       port map (
            A => A3,
            B => ONES3,
            C_in => FA2_C,
            S => S_3,
            C_out => CARRY);

    C_OUT<=CARRY;
    S0<=S_0;
    S1<=S_1;
    S2<=S_2;
    S3<=S_3;
    ZERO <=NOT S_0 AND NOT S_1 AND NOT S_2 AND NOT S_3 AND CARRY;

    OVERFLOW<= NOT(FA2_C AND (A3 XOR ONES3)) AND M;


    --OVERFLOW<= (FA2_C AND (A3 XOR ONES3)) XOR FA2_C;
    --OVERFLOW<= (C_OUT XOR FA2_C);
end Behavioral;
```

- *Instruction Decoder*

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 04.06.2023 21:03:54
-- Design Name:
-- Module Name: InstructionDecoder - Behavioral
-- Project Name:
-- Target Devices:
-- Tool Versions:
-- Description:
--
-- Dependencies:
```

```
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------------
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--use IEEE.NUMERIC_STD.ALL;

-- Uncomment the following library declaration if instantiating
-- any Xilinx leaf cells in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

entity InstructionDecoder is
   Port ( InsBus : in STD_LOGIC_VECTOR (11 downto 0);  --instruction bus
          RCJ : in STD_LOGIC_VECTOR (3 downto 0);      --Register check for jump
          RegEn : out STD_LOGIC_VECTOR (2 downto 0);   --Register Enable
          LS : out STD_LOGIC;                          --Load Select
          ImVal : out STD_LOGIC_VECTOR (3 downto 0);   --Immediate Value
          RegS1 : out STD_LOGIC_VECTOR (2 downto 0);   --Register Select 1
          RegS2 : out STD_LOGIC_VECTOR (2 downto 0);   --Register Select 2
          AddSubS : out STD_LOGIC;                     --Add/Sub Select
          JumpFlag : out STD_LOGIC;                    --JumpFlag
          ATJ : out STD_LOGIC_VECTOR (2 downto 0));    --Address to jump
end InstructionDecoder;

architecture Behavioral of InstructionDecoder is
SIGNAL op : std_logic_vector(1 downto 0);
signal jumpSel : std_logic := '0';
signal regA : std_logic_vector(2 downto 0);
signal regB : std_logic_vector(2 downto 0);
signal immediateValue_s : std_logic_vector(3 downto 0);
signal jumpAddress : std_logic_vector(2 downto 0);
signal regChkForJump : std_logic_vector(3 downto 0);
signal muxSelect_1 : std_logic_vector(2 downto 0);
signal muxSelect_2 : std_logic_vector(2 downto 0);
signal regEn_s : std_logic_vector(2 downto 0);
signal loadSel : std_logic;
signal addSub_s : std_logic;
signal immVal_s : std_logic_vector(3 downto 0);
begin
```

```
    op <= InsBus(11 downto 10);
    regA <= InsBus(9 downto 7);
    regB <= InsBus(6 downto 4);
    immediateValue_s <= InsBus(3 downto 0);
    regChkForJump <= RCJ;
    process (op, regA, regB, immediateValue_s, regChkForJump )

    begin
      jumpSel <= '0';
       if op = "00" then   --ADD---RA <- RA+RB
          muxSelect_1<=regA;
          muxSelect_2<=regB;
          addSub_s<='0';
          loadSel<='1';
          regEn_s<=regA; --Register enable
       elsif op = "01"then --NEG
          muxSelect_1<=regA;
          muxSelect_2<="000";
          addSub_s<='1';
          loadSel<='1';
          regEn_s<=regA;
       elsif op = "10" then --MOVI--
            --Register Enable
          immVal_s<=immediateValue_s;
          loadSel <='0';
          regEn_s<=regA;
       elsif op = "11" then   --JZR --
          muxSelect_1<=regA;
          jumpAddress<=immediateValue_s(2 downto 0);
          if RCJ= "0000" then   --PC<--D  regChkForJump
             jumpSel<='1';
          else               --PC<--PC+1
             jumpSel<='0';
          end if;
       end if;
    end process;

    JumpFlag <= jumpSel;
    RegEn <= regEn_s;
    LS <= loadSel;
    ImVal <= immVal_s;
    RegS1 <= muxSelect_1;
    RegS2 <= muxSelect_2;
    AddSubS <= addSub_s;
    ATJ <= jumpAddress;
end Behavioral;
```

- *Three Bit Adder*

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 15.03.2023 15:44:28
-- Design Name:
-- Module Name: RCA_4 - Behavioral
-- Project Name:
-- Target Devices:
-- Tool Versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------------


library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--use IEEE.NUMERIC_STD.ALL;

-- Uncomment the following library declaration if instantiating
-- any Xilinx leaf cells in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

entity RCA_4 is
   Port ( A : in STD_LOGIC_VECTOR(2 DOWNTO 0);
          B : in STD_LOGIC_VECTOR(2 DOWNTO 0);
          C_in : in STD_LOGIC;
          S : out STD_LOGIC_VECTOR(2 DOWNTO 0);
          C_out : out STD_LOGIC);
```

```
end RCA_4;

architecture Behavioral of RCA_4 is

   component FA
      port (
      A: in std_logic;
      B: in std_logic;
      C_in: in std_logic;
      S: out std_logic;
      C_out: out std_logic);
   end component;

   SIGNAL FA0_C,FA1_C,FA2_C:  std_logic;
   SIGNAL TEMP_S:STD_LOGIC_VECTOR(2 DOWNTO 0);

begin
   FA_0 : FA
      port map (
          A => A(0),
          B => B(0),
          C_in => '0',
          S => TEMP_S(0),
          C_Out => FA0_C);
   FA_1 : FA
      port map (
          A => A(1),
          B => B(1),
          C_in => FA0_C,
          S => TEMP_S(1),
          C_Out => FA1_C);
    FA_2 : FA
       port map (
          A => A(2),
          B => B(2),
          C_in => FA1_C,
          S => TEMP_S(2),
          C_Out => FA2_C);


   S<=TEMP_S;
   C_out<=FA2_C;
end Behavioral;
```

- *Program Rom*

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 04.06.2023 19:07:16
-- Design Name:
-- Module Name: ProgramROM - Behavioral
-- Project Name:
-- Target Devices:
-- Tool Versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------------


library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use ieee.numeric_std.all;
-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--use IEEE.NUMERIC_STD.ALL;

-- Uncomment the following library declaration if instantiating
-- any Xilinx leaf cells in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

entity ProgramROM is
   Port ( address : in STD_LOGIC_VECTOR (2 downto 0);
          data : out STD_LOGIC_VECTOR (11 downto 0));
end ProgramROM;

architecture Behavioral of ProgramROM is

type rom_type is array (0 to 7) of std_logic_vector(11 downto 0);
signal rom : rom_type :=(
   "101110000001", --0    MOVI R7,1
   "101100000010", --1    MOVI R6,2
```

```
   "101010000011", --2    MOVI R5,3
   "001111100000", --3    ADD R7 <- R7 + R6
   "001111010000", --4    ADD R7 <- R7 + R5
   "110000000000", --5    JZR R0,0
   "000000000000", --6
   "000000000000"  --7
  );
begin
   data <= rom(to_integer(unsigned(address)));
end Behavioral;
```

- *Slow Clock*

```
----------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 04/04/2023 03:09:16 PM
-- Design Name:
-- Module Name: Slow_Clk - Behavioral
-- Project Name:
-- Target Devices:
-- Tool Versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------


library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
entity Slow_Clk is
   Port ( Clk_in : in STD_LOGIC;
          Clk_out : out STD_LOGIC);
end Slow_Clk;

architecture Behavioral of Slow_Clk is
```

```
signal count : integer := 0;
signal clk_status : std_logic := '0';

begin
   process(Clk_in) begin
      if(rising_edge(Clk_in)) then
         count <= count + 1;
         if(count = 20) then
            clk_status <= NOT clk_status;
            Clk_out <= clk_status;
            count <= 1;
         end if;
      end if;
   end process;
end Behavioral;
```

- *Program Counter*

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 05.06.2023 13:07:48
-- Design Name:
-- Module Name: ProgramCounter_3Bit - Behavioral
-- Project Name:
-- Target Devices:
-- Tool Versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------------


library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
```

```
--use IEEE.NUMERIC_STD.ALL;

-- Uncomment the following library declaration if instantiating
-- any Xilinx leaf cells in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

entity ProgramCounter_3Bit is
   Port ( Reset_PushButton : in STD_LOGIC;
          Clk : in STD_LOGIC;
          pc_in : in STD_LOGIC_VECTOR (2 downto 0);
          MemSel : out STD_LOGIC_VECTOR (2 downto 0));
end ProgramCounter_3Bit;

architecture Behavioral of ProgramCounter_3Bit is
begin
   process(Clk)
   begin
   if(rising_edge(Clk)) then
      if(Reset_PushButton='1') then
        MemSel<="000";
      else
        MemSel<=pc_in;
      end if;
   end if;
end process;


end Behavioral;
```

- *D Flip Flop*

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 04/08/2023 02:57:21 PM
-- Design Name:
-- Module Name: D_FF - Behavioral
-- Project Name:
-- Target Devices:
-- Tool Versions:
-- Description:
--
-- Dependencies:
```

```
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------------


library IEEE;
use IEEE.STD_LOGIC_1164.ALL;


entity D_FF is
    Port ( D : in STD_LOGIC;
           Res : in STD_LOGIC;
           Clk : in STD_LOGIC;
           Q : out STD_LOGIC
           --Qbar : out STD_LOGIC
           );
end D_FF;

architecture Behavioral of D_FF is
begin
    process (Clk) begin
        if(rising_edge (Clk)) then
            if Res = '1' then
                Q <= '0';
                --Qbar <= '1' ;
            else
                Q <= D;
                --Qbar <= not D;
             end if;
        end if;
     end process;




end Behavioral;
```

4)All Simulation files

- *Nano Processor Simulation Code*

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 07.06.2023 18:29:37
-- Design Name:
-- Module Name: SimNanoProcessor - Behavioral
-- Project Name:
-- Target Devices:
-- Tool Versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------------


library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--use IEEE.NUMERIC_STD.ALL;

-- Uncomment the following library declaration if instantiating
-- any Xilinx leaf cells in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

entity SimNanoProcessor is
--  Port ( );
end SimNanoProcessor;

architecture Behavioral of SimNanoProcessor is
COMPONENT NanoProcessor is
   Port ( pushButton : in STD_LOGIC;
          jump_Flag: out std_logic;
          Address_to_jump:out std_logic_vector(2 downto 0);
          Clk : in STD_LOGIC;
          Imm_Value:out std_logic_vector(3 downto 0);
          LED : out STD_LOGIC_VECTOR (3 downto 0);
```

```
        LED_R6 : out std_logic_vector(3 downto 0);
        LED_R5 : out std_logic_vector(3 downto 0);
        nextInsVal : out std_logic_vector(2 downto 0);
        LED_OVERFLOW:out STD_LOGIC;
        RegisterEnable:out std_logic_vector(2 downto 0);
        RegisterBank_DataIn: out std_logic_vector(3 downto 0);
        LED_ZERO:out STD_LOGIC;
        Instruction : out std_logic_vector(11 downto 0);
        Register_check_for_jump:out std_logic_vector(3 downto 0);
        LED_R0:out std_logic_vector(3 downto 0)
        );
end COMPONENT;
SIGNAL PUSHBUTTON,LED_OVERFLOW,LED_ZERO:STD_LOGIC;
SIGNAL LED:STD_LOGIC_VECTOR(3 DOWNTO 0);

signal LED_R6 ,ImmValue: std_logic_vector(3 downto 0);
signal LED_R5 : std_logic_vector(3 downto 0);
SIGNAL CLK: STD_LOGIC:='1';
signal nextInsVal : std_logic_vector(2 downto 0);
SIGNAL Register_Enable :std_logic_vector(2 downto 0);
Signal Register_Bank_Data_In:std_logic_vector(3 downto 0);
signal Instruction : std_logic_vector(11 downto 0);
signal jump_Flagg: std_logic;
signal Address_to_jumpp:std_logic_vector(2 downto 0);
signal Register_check_for_jumpp,LED_R0:std_logic_vector(3 downto 0);
begin
 UUT:NanoProcessor Port Map(
   pushButton=>PUSHBUTTON,
   Clk=>CLK,
   RegisterEnable=>Register_Enable,
   RegisterBank_DataIn=>Register_Bank_Data_In,
   LED=>LED,
   LED_R6 => LED_R6,
   LED_R5 => LED_R5,
   nextInsVal => nextInsVal,
   LED_OVERFLOW=>LED_OVERFLOW,
   Imm_Value=>ImmValue,
   LED_ZERO=>LED_ZERO,
   Instruction => Instruction,
   jump_Flag=>jump_Flagg,
   Address_to_jump=>Address_to_jumpp,
   Register_check_for_jump=>Register_check_for_jumpp,
   LED_R0=>LED_R0

 );
 PROCESS BEGIN
```

```
   WAIT FOR 30 NS;
   CLK <= NOT CLK;
 END PROCESS;
 PROCESS BEGIN
   wait for 45 ns;
   PUSHBUTTON<='1';
   wait for 60 ns;
   PUSHBUTTON<='0';
--   wait for 200 ns;
--   PUSHBUTTON<='1';
--   wait for 50 ns;
--   PUSHBUTTON<='0';
   WAIT;
 END PROCESS;

end Behavioral;
```

- *Add Sub Unit Simulation Code*

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 05.06.2023 11:36:58
-- Design Name:
-- Module Name: SimAddSubUnit - Behavioral
-- Project Name:
-- Target Devices:
-- Tool Versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------------


library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

-- Uncomment the following library declaration if using
```

```
-- arithmetic functions with Signed or Unsigned values
--use IEEE.NUMERIC_STD.ALL;

-- Uncomment the following library declaration if instantiating
-- any Xilinx leaf cells in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

entity SimAddSubUnit is
--  Port ( );
end SimAddSubUnit;

architecture Behavioral of SimAddSubUnit is
component AddSubUnit
   Port ( A0 : in STD_LOGIC;
          A1 : in STD_LOGIC;
          A2 : in STD_LOGIC;
          A3 : in STD_LOGIC;
          M : in STD_LOGIC;
          B0 : in STD_LOGIC;
          B1 : in STD_LOGIC;
          B2 : in STD_LOGIC;
          B3 : in STD_LOGIC;
          S0 : out STD_LOGIC;
          S1 : out STD_LOGIC;
          S2 : out STD_LOGIC;
          S3 : out STD_LOGIC;
          C_OUT : out STD_LOGIC;
          ZERO: out STD_LOGIC;
          OVERFLOW: out STD_LOGIC );
end component;
SIGNAL A0,A1,A2,A3,M,B0,B1,B2,B3,S0,S1,S2,S3,C_OUT,OVERFLOW,zero:
STD_LOGIC;
begin
UUT : AddSubUnit
   port map(
      A0=>A0,
      A1=>A1,
      A2=>A2,
      A3=>A3,
      M=>M,
      B0=>B0,
      B1=>B1,
      B2=>B2,
      B3=>B3,
      S0=>S0,
```

```
        S1=>S1,
        S2=>S2,
        S3=>S3,
        C_OUT=>C_OUT,
        OVERFLOW=>OVERFLOW,
        ZERO=>zero);
process
   begin

        A0 <= '0'; -- set initial values
        A1 <= '0';
        A2 <= '0';
        A3 <= '0';
        B0 <= '0'; -- set initial values
        B1 <= '1';
        B2 <= '1';
        B3 <= '1';
        M<='0';

    WAIT FOR 100 ns; -- after 100 ns change inputs

        A0 <= '1'; -- set initial values
        A1 <= '0';
        A2 <= '1';
        A3 <= '0';
        B0 <= '1'; -- set initial values
        B1 <= '1';
        B2 <= '0';
        B3 <= '0';

    WAIT FOR 100 ns; -- after 100 ns change inputs

        A0 <= '1'; -- set initial values
        A1 <= '0';
        A2 <= '1';
        A3 <= '0';
        B0 <= '1'; -- set initial values
        B1 <= '1';
        B2 <= '0';
        B3 <= '1';

    WAIT FOR 100 ns; -- after 100 ns change inputs

        A0 <= '1'; -- set initial values
        A1 <= '1';
        A2 <= '1';
```

```
        A3 <= '1';
        B0 <= '1'; -- set initial values
        B1 <= '1';
        B2 <= '1';
        B3 <= '0';

     WAIT FOR 100 ns; -- after 100 ns change inputs

         A0 <= '0'; -- set initial values
         A1 <= '0';
         A2 <= '0';
         A3 <= '1';
         B0 <= '0'; -- set initial values
         B1 <= '0';
         B2 <= '0';
         B3 <= '1';

    WAIT; -- will wait forever

end process;

end Behavioral;
```

- *Sim Instruction Decoder*

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 05.06.2023 01:00:42
-- Design Name:
-- Module Name: SimInstructionDecoder - Behavioral
-- Project Name:
-- Target Devices:
-- Tool Versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
```

```
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------------


library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--use IEEE.NUMERIC_STD.ALL;

-- Uncomment the following library declaration if instantiating
-- any Xilinx leaf cells in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

entity SimInstructionDecoder is
--  Port ( );
end SimInstructionDecoder;

architecture Behavioral of SimInstructionDecoder is
component InstructionDecoder
    Port ( InsBus : in STD_LOGIC_VECTOR (11 downto 0);  --instruction bus
    RCJ : in STD_LOGIC_VECTOR (3 downto 0);      --Register check for jump
    RegEn : out STD_LOGIC_VECTOR (2 downto 0);   --Register Enable
    LS : out STD_LOGIC;                          --Load Select
    ImVal : out STD_LOGIC_VECTOR (3 downto 0);   --Immediate Value
    RegS1 : out STD_LOGIC_VECTOR (2 downto 0);   --Register Select 1
    RegS2 : out STD_LOGIC_VECTOR (2 downto 0);   --Register Select 2
    AddSubS : out STD_LOGIC;                     --Add/Sub Select
    JumpFlag : out STD_LOGIC;                    --JumpFlag
    ATJ : out STD_LOGIC_VECTOR (2 downto 0));    --Address to jump
end component;
SIGNAL insbus : STD_LOGIC_VECTOR(11 downto 0);
SIGNAL rcj    :STD_LOGIC_VECTOR(3 downto 0);
SIGNAL regen  :STD_LOGIC_VECTOR(2 downto 0);
SIGNAL ls : STD_LOGIC;                           --Load Select
SIGNAL imval :STD_LOGIC_VECTOR (3 downto 0);     --Immediate Value
SIGNAL regs1 :STD_LOGIC_VECTOR (2 downto 0);     --Register Select 1
SIGNAL regs2 :STD_LOGIC_VECTOR (2 downto 0);     --Register Select 2
SIGNAL addsubs : STD_LOGIC;                      --Add/Sub Select
SIGNAL jumpflag : STD_LOGIC;                     --JumpFlag
SIGNAL atj : STD_LOGIC_VECTOR (2 downto 0);
begin
```

```
UUT :InstructionDecoder
   port map(
      insbus=>InsBus,
      rcj=>RCJ,
      regen=>RegEn,
      ls=>LS,
      imval=>ImVal,
      regs1=>RegS1,
      regs2=>RegS2,
      addsubs=>AddSubS,
      jumpflag=>JumpFlag,
      atj=>ATJ);
process
 begin
   rcj<="0000";
   insbus<="101110000001"; --MOVI
   WAIT FOR 100ns;
   insbus<="101100000010";  --MOVI
   WAIT FOR 100ns;
   insbus<="101010000011"; -- MOVI
   WAIT FOR 100ns;
   insbus<="001111100000"; -- ADD
   WAIT FOR 100ns;
   insbus<="110000000000";  --JZR
   WAIT FOR 100ns;
   insbus<="110000000000"; --JZR
   WAIT FOR 100ns;
   insbus<="000000000000";
   WAIT;
end process;
end Behavioral;
```

- *8 Way 4 Bit Multiplexer Simulation*

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 05.06.2023 09:15:06
-- Design Name:
-- Module Name: SimMul_8_Way_4_Bit - Behavioral
-- Project Name:
-- Target Devices:
```

```
-- Tool Versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------------


library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--use IEEE.NUMERIC_STD.ALL;

-- Uncomment the following library declaration if instantiating
-- any Xilinx leaf cells in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

entity SimMul_8_Way_4_Bit is
--  Port ( );
end SimMul_8_Way_4_Bit;

architecture Behavioral of SimMul_8_Way_4_Bit is
component Mux_8_Way_4_Bit is
   Port ( Reg_Sel : in STD_LOGIC_VECTOR (2 downto 0);
          R0 : in STD_LOGIC_VECTOR (3 downto 0);
          R1 : in STD_LOGIC_VECTOR (3 downto 0);
          R2 : in STD_LOGIC_VECTOR (3 downto 0);
          R3 : in STD_LOGIC_VECTOR (3 downto 0);
          R4 : in STD_LOGIC_VECTOR (3 downto 0);
          R5 : in STD_LOGIC_VECTOR (3 downto 0);
          R6 : in STD_LOGIC_VECTOR (3 downto 0);
          R7 : in STD_LOGIC_VECTOR (3 downto 0);
          Mux_Out  : out STD_LOGIC_VECTOR (3 downto 0));
end component;
signal Reg_Sel :STD_LOGIC_VECTOR (2 downto 0);
signal R0 : STD_LOGIC_VECTOR (3 downto 0);
signal R1 : STD_LOGIC_VECTOR (3 downto 0);
signal R2 : STD_LOGIC_VECTOR (3 downto 0);
signal R3 : STD_LOGIC_VECTOR (3 downto 0);
```

```
signal R4 : STD_LOGIC_VECTOR (3 downto 0);
signal R5 : STD_LOGIC_VECTOR (3 downto 0);
signal R6 : STD_LOGIC_VECTOR (3 downto 0);
signal R7 : STD_LOGIC_VECTOR (3 downto 0);
signal Mux_Out  :STD_LOGIC_VECTOR (3 downto 0);

begin
UUT:Mux_8_Way_4_Bit port map(
   Reg_Sel=>Reg_Sel,
   R0=>R0,
   R1=>R1,
   R2=>R2,
   R3=>R3,
   R4=>R4,
   R5=>R5,
   R6=>R6,
   R7=>R7,
   Mux_Out=>Mux_Out);
process
   begin
      R0<="0000";
      R1<="0001";
      R2<="0010";
      R3<="0011";
      R4<="0100";
      R5<="0101";
      R6<="0110";
      R7<="0111";
      Reg_Sel<="000";
      WAIT FOR 100 ns;
      Reg_Sel<="001";
      WAIT FOR 100 ns;
      Reg_Sel<="010";
      WAIT FOR 100 ns;
      Reg_Sel<="011";
      WAIT FOR 100 ns;
      Reg_Sel<="100";
      WAIT FOR 100 ns;
      Reg_Sel<="101";
      WAIT FOR 100 ns;
      Reg_Sel<="110";
      WAIT FOR 100 ns;
      Reg_Sel<="111";
      WAIT;
end process ;
end Behavioral;
```

- *Program Rom Simulation Code*

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 04.06.2023 19:40:23
-- Design Name:
-- Module Name: SimProgramROM - Behavioral
-- Project Name:
-- Target Devices:
-- Tool Versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------------


library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

-- Uncomment the following library declaration if using
```

```
-- arithmetic functions with Signed or Unsigned values
--use IEEE.NUMERIC_STD.ALL;

-- Uncomment the following library declaration if instantiating
-- any Xilinx leaf cells in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

entity SimProgramROM is
--  Port ( );
end SimProgramROM;

architecture Behavioral of SimProgramROM is
component ProgramROM
   port(
   address : in std_logic_vector(2 downto 0);
   data : out std_logic_vector(11 downto 0)
   );
end component;
SIGNAL temp_address :std_logic_vector(2 downto 0);
SIGNAL temp_data: std_logic_vector(11 downto 0);
begin
   UUT:ProgramROM port map(
      address=>temp_address,
      data=>temp_data
      );
process
 begin
   temp_address<="000";
```

```
    WAIT FOR 100 ns;
    temp_address<="001";
    WAIT FOR 100 ns;
    temp_address<="010";
    WAIT FOR 100 ns;
    temp_address<="011";
    WAIT FOR 100 ns;
    temp_address<="100";
    WAIT FOR 100 ns;
    temp_address<="101";
    WAIT FOR 100 ns;
    temp_address<="110";
    WAIT FOR 100 ns;
    temp_address<="111";
    WAIT;

end process;
end Behavioral;
```

- *RCA Simulation Code*

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 15.03.2023 16:01:59
-- Design Name:
-- Module Name: TB_4_RCA - Behavioral
-- Project Name:
-- Target Devices:
```

```
-- Tool Versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------------


library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--use IEEE.NUMERIC_STD.ALL;

-- Uncomment the following library declaration if instantiating
-- any Xilinx leaf cells in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

entity TB_4_RCA is
--  Port ( );
end TB_4_RCA;

architecture Behavioral of TB_4_RCA is
```

```
COMPONENT RCA_4 is
   Port ( A : in STD_LOGIC_VECTOR(2 DOWNTO 0);
          B : in STD_LOGIC_VECTOR(2 DOWNTO 0);
          C_in : in STD_LOGIC;
          S : out STD_LOGIC_VECTOR(2 DOWNTO 0);
          C_out : out STD_LOGIC);
end COMPONENT;
SIGNAL A,B,S : std_logic_vector(2 downto 0);
SIGNAL C_out,C_in : std_logic;
begin
 UUT: RCA_4 PORT MAP(
   A=>A,
   B=>B,
   C_in=>C_in,
   C_out=>C_out,
   S=>S
 );
 process
 begin
     A<="000";
     B<="001";
     C_in<='0';

     WAIT FOR 100 ns; -- after 100 ns change inputs
     A<="001";

     WAIT FOR 100 ns; -- after 100 ns change inputs
     A<="010";
```

```
    WAIT FOR 100 ns; -- after 100 ns change inputs
    A<="011";

    WAIT FOR 100 ns; -- after 100 ns change inputs
    A<="100";

    WAIT FOR 100 ns; -- after 100 ns change inputs

    A<="101";
    WAIT FOR 100 ns; -- after 100 ns change inputs

    A<="110";
    WAIT FOR 100 ns; -- after 100 ns change inputs

    A<="111";
--    WAIT FOR 100 ns; -- after 100 ns change inputs
--    B<="001";
--    A<="000";
    WAIT; -- will wait forever
  end process;

end Behavioral;
```

- *D Flip Flop Simulation*

```
----------------------------------------------------------------------------
-- Company:
-- Engineer:
--
```

```
-- Create Date: 04/08/2023 02:57:21 PM
-- Design Name:
-- Module Name: D_FF - Behavioral
-- Project Name:
-- Target Devices:
-- Tool Versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------------


library IEEE;
use IEEE.STD_LOGIC_1164.ALL;



entity D_FF is
   Port ( D : in STD_LOGIC;
          Res : in STD_LOGIC;
          Clk : in STD_LOGIC;
          Q : out STD_LOGIC
          --Qbar : out STD_LOGIC
          );
end D_FF;
```

```
architecture Behavioral of D_FF is
begin
   process (Clk) begin
      if(rising_edge (Clk)) then
         if Res = '1' then
               Q <= '0';
               --Qbar <= '1' ;
         else
               Q <= D;
               --Qbar <= not D;
          end if;
       end if;
    end process;




end Behavioral;
```

SIM REGISTER BANK

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 11.06.2023 20:36:09
-- Design Name:
-- Module Name: SimRegisterBank - Behavioral
-- Project Name:
```

```
-- Target Devices:
-- Tool Versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------------


library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--use IEEE.NUMERIC_STD.ALL;

-- Uncomment the following library declaration if instantiating
-- any Xilinx leaf cells in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

entity SimRegisterBank is
--  Port ( );
end SimRegisterBank;
```

```
architecture Behavioral of SimRegisterBank is
component Reg_Bank is
   Port ( Clk : in STD_LOGIC;
          Reg_EN : in STD_LOGIC_VECTOR (2 downto 0);
          Reg_Bank_In : in STD_LOGIC_VECTOR (3 downto 0);
          push_button:in std_logic;  ---push_button=1 =>reset
          Reg_B_Out_0 : out STD_LOGIC_VECTOR (3 downto 0);
          Reg_B_Out_1 : out STD_LOGIC_VECTOR (3 downto 0);
          Reg_B_Out_2 : out STD_LOGIC_VECTOR (3 downto 0);
          Reg_B_Out_3 : out STD_LOGIC_VECTOR (3 downto 0);
          Reg_B_Out_4 : out STD_LOGIC_VECTOR (3 downto 0);--register banks
          Reg_B_Out_5 : out STD_LOGIC_VECTOR (3 downto 0);
          Reg_B_Out_6 : out STD_LOGIC_VECTOR (3 downto 0);
          Reg_B_Out_7 : out STD_LOGIC_VECTOR (3 downto 0)
          );
end component;
signal pushButton:std_logic;
signal clock:std_logic;
signal reg_enable:std_logic_vector(2  downto 0);
signal reg_in,R0,R1,R2,R3,R4,R5,R6,R7:std_logic_vector(3  downto 0);
begin
UUT: Reg_Bank port map(
   Clk=>clock,
   Reg_En=>reg_enable,
   Reg_Bank_In=>reg_in,
   push_button=>pushButton,
   Reg_B_Out_0=>R0,
   Reg_B_Out_1=>R1,
   Reg_B_Out_2=>R2,
```

```
    Reg_B_Out_3=>R3,
    Reg_B_Out_4=>R4,
    Reg_B_Out_5=>R5,
    Reg_B_Out_6=>R6,
    Reg_B_Out_7=>R7
    );
process begin
    pushButton<='1';
    ------------------------------
    wait for 100 ns;
    pushButton<='0';

    clock<='0';
    reg_enable<="111";
    reg_in<="0001";
    wait for 100 ns;
    clock<='1';

    wait for 100 ns;
    clock<='0';
    reg_enable<="110";
    reg_in<="0010";
    wait for 100 ns;
    clock<='1';

    wait for 100 ns;
    clock<='0';
    reg_enable<="101";
    reg_in<="0011";
```

```
    wait for 100 ns;
    clock<='1';

    wait for 100 ns;
    clock<='0';
    reg_enable<="011";
    reg_in<="0101";
    wait for 100 ns;
    clock<='1';



 end process;

end Behavioral;
```

SIM ADD SUB UNIT

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 05.06.2023 11:36:58
-- Design Name:
-- Module Name: SimAddSubUnit - Behavioral
-- Project Name:
-- Target Devices:
-- Tool Versions:
-- Description:
--
```

```
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------------


library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--use IEEE.NUMERIC_STD.ALL;

-- Uncomment the following library declaration if instantiating
-- any Xilinx leaf cells in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

entity SimAddSubUnit is
--  Port ( );
end SimAddSubUnit;

architecture Behavioral of SimAddSubUnit is
component AddSubUnit
   Port ( A0 : in STD_LOGIC;
          A1 : in STD_LOGIC;
```

```
        A2 : in STD_LOGIC;
        A3 : in STD_LOGIC;
        M : in STD_LOGIC;
        B0 : in STD_LOGIC;
        B1 : in STD_LOGIC;
        B2 : in STD_LOGIC;
        B3 : in STD_LOGIC;
        S0 : out STD_LOGIC;
        S1 : out STD_LOGIC;
        S2 : out STD_LOGIC;
        S3 : out STD_LOGIC;
        C_OUT : out STD_LOGIC;
        ZERO: out STD_LOGIC;
        OVERFLOW: out STD_LOGIC );
end component;
SIGNAL A0,A1,A2,A3,M,B0,B1,B2,B3,S0,S1,S2,S3,C_OUT,OVERFLOW,zero:  STD_LOGIC;
begin
UUT : AddSubUnit
   port map(
      A0=>A0,
      A1=>A1,
      A2=>A2,
      A3=>A3,
      M=>M,
      B0=>B0,
      B1=>B1,
      B2=>B2,
      B3=>B3,
      S0=>S0,
```

```
        S1=>S1,
        S2=>S2,
        S3=>S3,
        C_OUT=>C_OUT,
        OVERFLOW=>OVERFLOW,
        ZERO=>zero);
process
   begin

      A0 <= '0'; -- set initial values
      A1 <= '0';
      A2 <= '0';
      A3 <= '0';
      B0 <= '0'; -- set initial values
      B1 <= '1';
      B2 <= '1';
      B3 <= '1';
      M<='0';

   WAIT FOR 100 ns; -- after 100 ns change inputs

      A0 <= '1'; -- set initial values
      A1 <= '0';
      A2 <= '1';
      A3 <= '0';
      B0 <= '1'; -- set initial values
      B1 <= '1';
      B2 <= '0';
      B3 <= '0';
```

```
    M<='1';
WAIT FOR 100 ns; -- after 100 ns change inputs

        A0 <= '1'; -- set initial values
        A1 <= '0';
        A2 <= '1';
        A3 <= '0';
        B0 <= '1'; -- set initial values
        B1 <= '1';
        B2 <= '0';
        B3 <= '0';
         M<='0';
  WAIT FOR 100 ns; -- after 100 ns change inputs

                A0 <= '1'; -- set initial values
                A1 <= '0';
                A2 <= '1';
                A3 <= '0';
                B0 <= '1'; -- set initial values
                B1 <= '1';
                B2 <= '0';
                B3 <= '0';
                 M<='1';

WAIT FOR 100 ns; -- after 100 ns change inputs

    A0 <= '1'; -- set initial values
    A1 <= '0';
    A2 <= '1';
```

```
A3 <= '0';
B0 <= '1'; -- set initial values
B1 <= '1';
B2 <= '0';
B3 <= '1';

WAIT FOR 100 ns; -- after 100 ns change inputs

A0 <= '1'; -- set initial values
A1 <= '1';
A2 <= '1';
A3 <= '1';
B0 <= '1'; -- set initial values
B1 <= '1';
B2 <= '1';
B3 <= '0';
M<='0';

WAIT FOR 100 ns; -- after 100 ns change inputs

A0 <= '0'; -- set initial values
A1 <= '0';
A2 <= '0';
A3 <= '1';
B0 <= '0'; -- set initial values
B1 <= '0';
B2 <= '0';
B3 <= '1';
```

```
   WAIT; -- will wait forever


end process;


end Behavioral;
```

SIM PROGRAM COUNTER

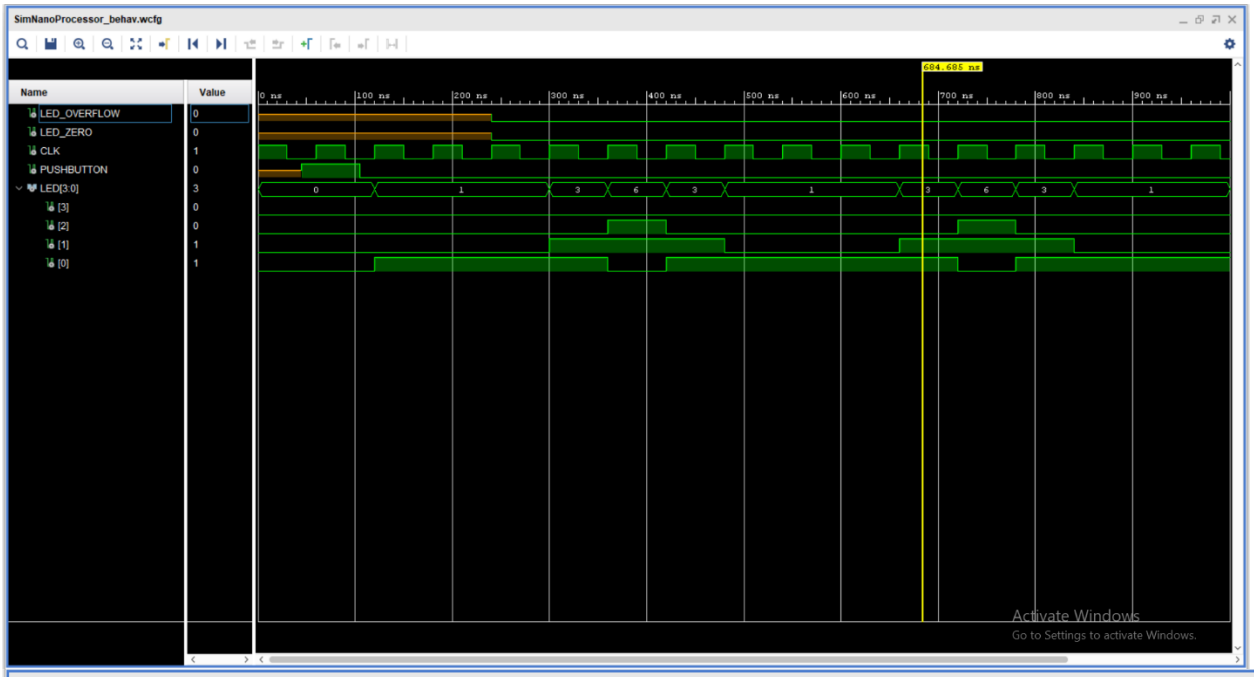
```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date: 11.06.2023 20:01:19
-- Design Name:
-- Module Name: SimProgramCounter - Behavioral
-- Project Name:
-- Target Devices:
-- Tool Versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
----------------------------------------------------------------------------------


library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

-- Uncomment the following library declaration if using
-- arithmetic functions with Signed or Unsigned values
--use IEEE.NUMERIC_STD.ALL;

-- Uncomment the following library declaration if instantiating
-- any Xilinx leaf cells in this code.
--library UNISIM;
--use UNISIM.VComponents.all;

entity SimProgramCounter is
--  Port ( );
```

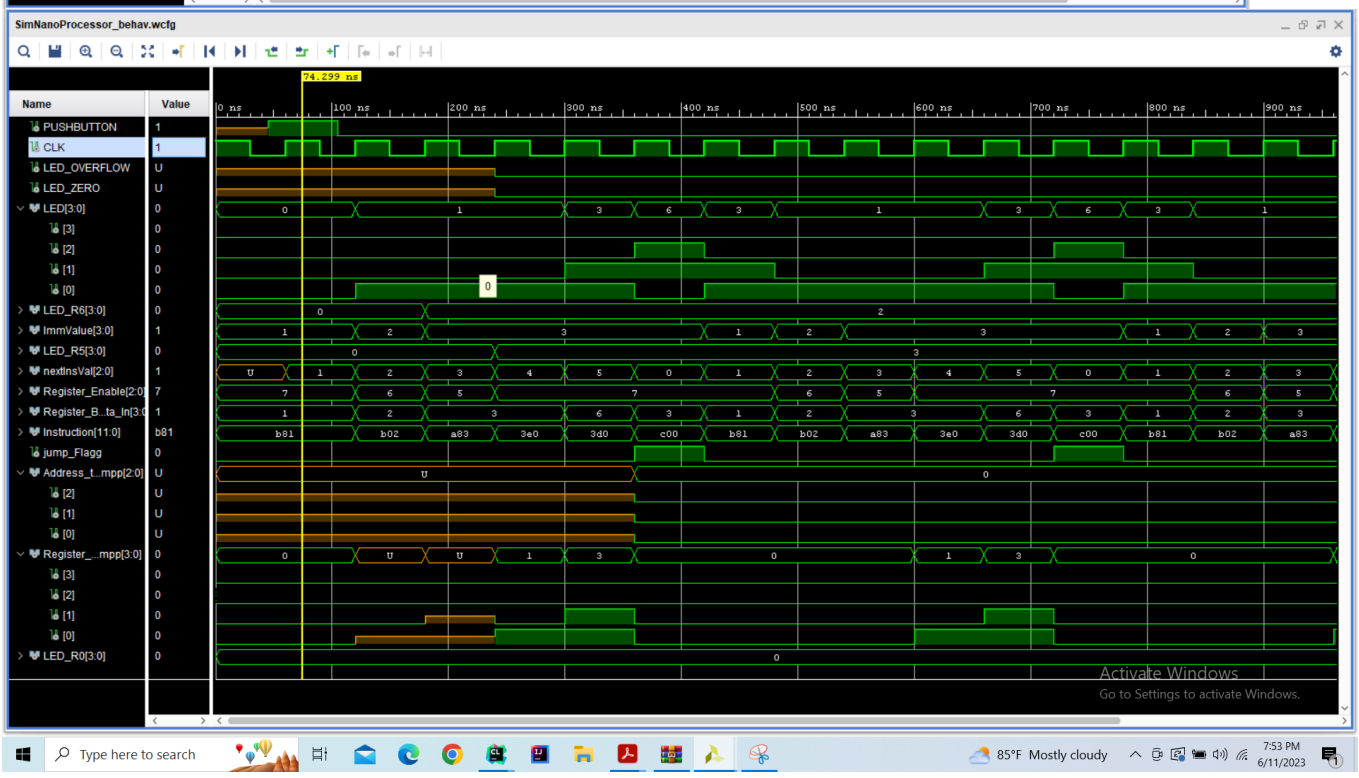
```
end SimProgramCounter;

architecture Behavioral of SimProgramCounter is
component ProgramCounter_3Bit is
    Port ( Reset_PushButton : in STD_LOGIC;
           Clk : in STD_LOGIC;
           pc_in : in STD_LOGIC_VECTOR  (2 downto 0);
           MemSel : out STD_LOGIC_VECTOR  (2 downto 0));
end component;
SIGNAL ResetPushButton:std_logic;
SIGNAL Clock:std_logic;
SIGNAL pcIn:std_logic_vector(2 downto 0);
SIGNAL Mem_Sel:std_logic_vector(2 downto 0);
begin
UUT:ProgramCounter_3Bit port map(
    Reset_PushButton=>ResetPushButton,
    Clk=>Clock,
    pc_in=>pcIn,
    MemSel=>Mem_Sel
);
process begin
    ResetPushButton<='1';
    Clock<='1';
    pcIn<="111";
    WAIT FOR 100 ns;
    Clock<='0';
    ResetPushButton<='0';

    pcIn<="101";
    WAIT FOR 100 ns;
    Clock<='1';
    ResetPushButton<='1';
    pcIn<="001";
    WAIT FOR 100 ns;
    Clock<='0';
    ResetPushButton<='0';
    pcIn<="011";
    WAIT FOR 100 ns;
    Clock<='1';
    pcIn<="110";
    WAIT FOR 100 ns;
    Clock<='0';
    pcIn<="111";
    WAIT FOR 100 ns;
    Clock<='1';
    pcIn<="100";
```

WAIT;
end process;
end Behavioral;

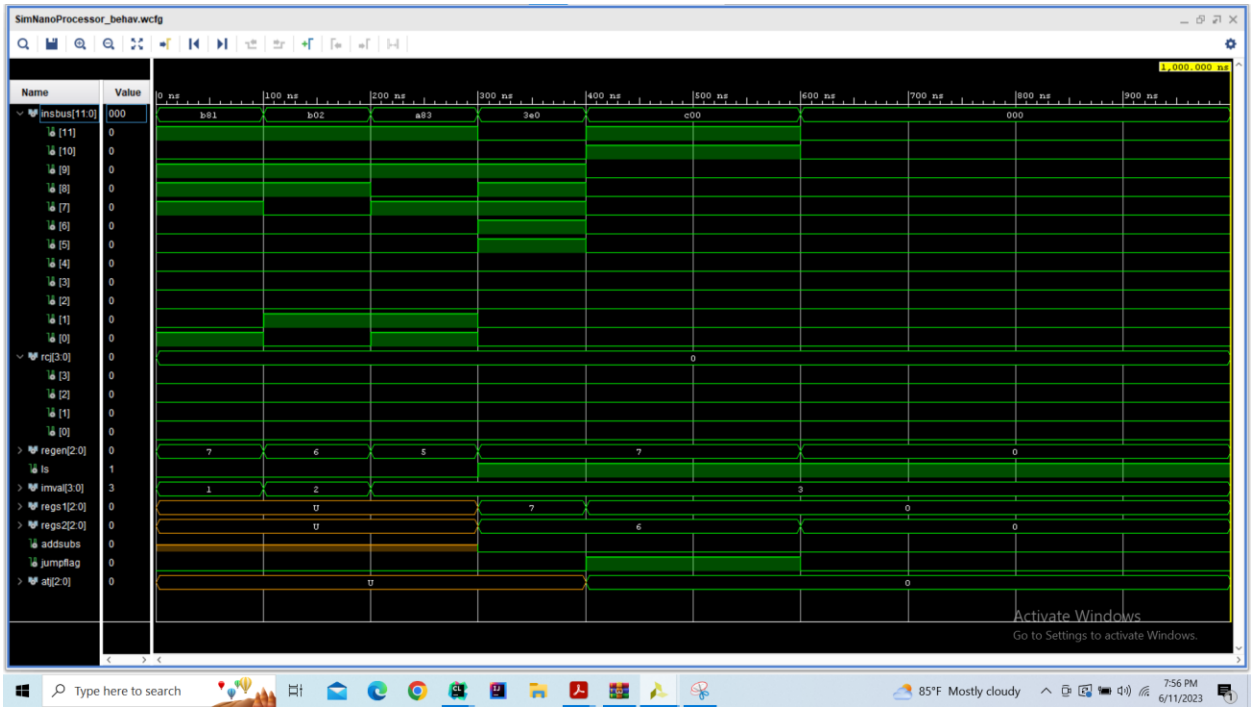
4) All timing diagrams

- **Nano Processor**

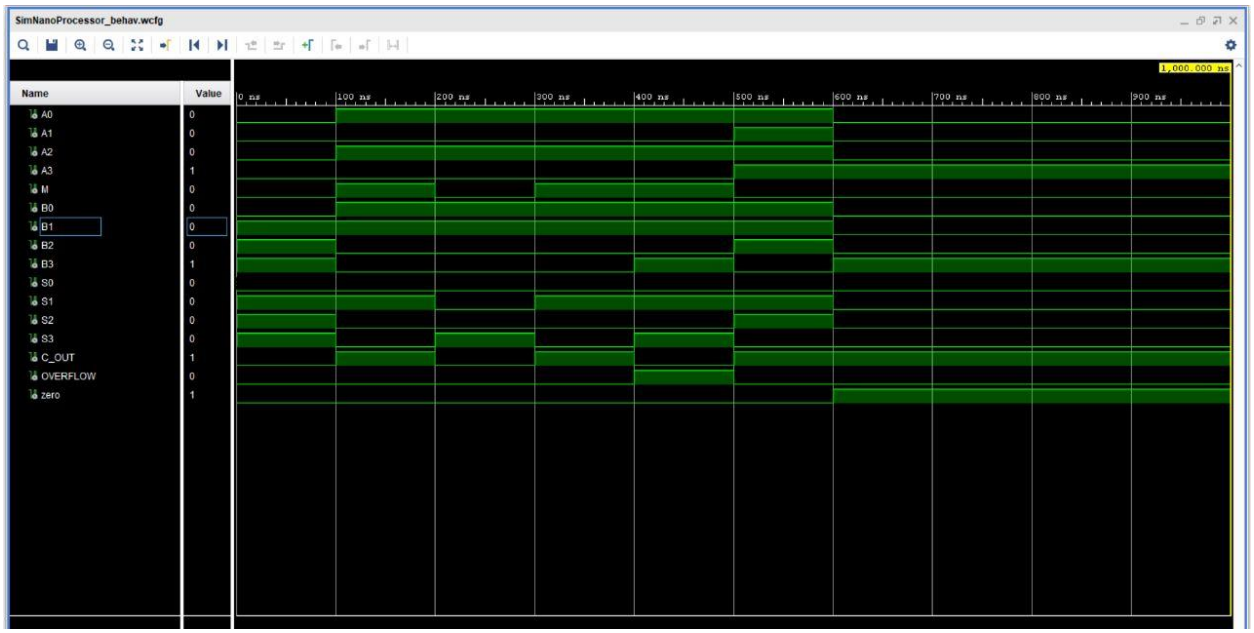




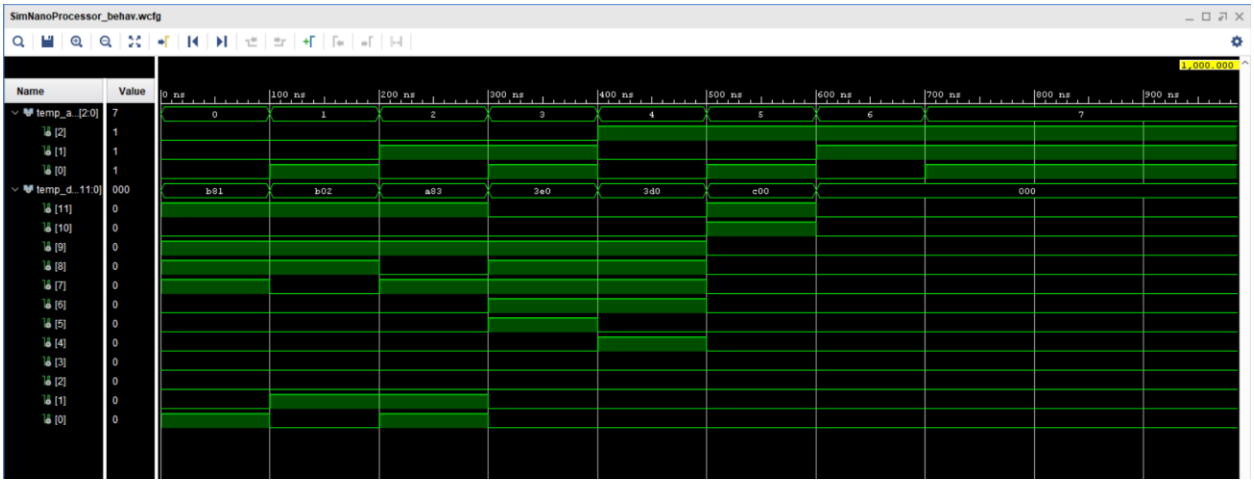
- **Instruction Decoder**



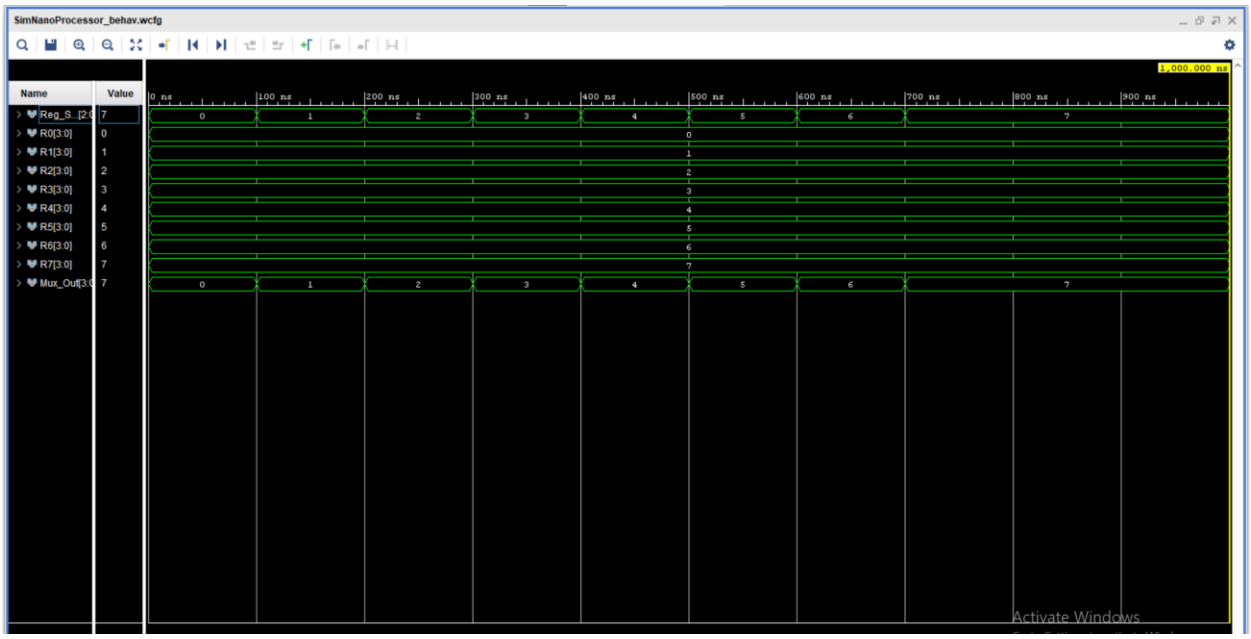
- **Add Sub Unit Timing Diagram**



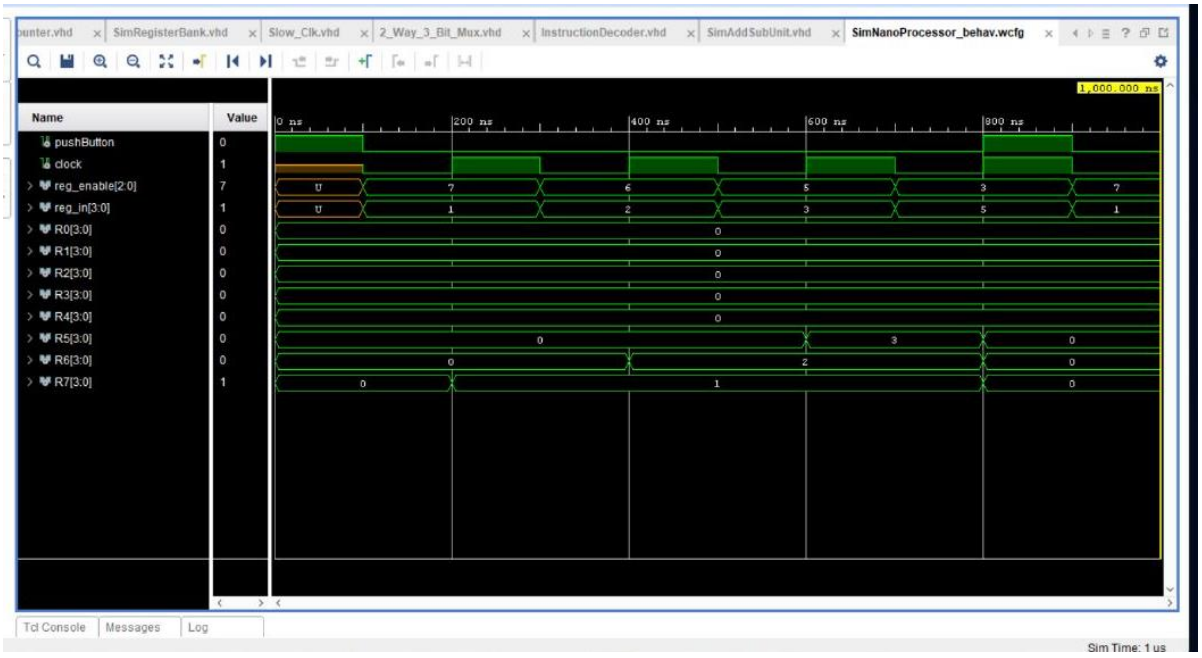
- **Program Rom**



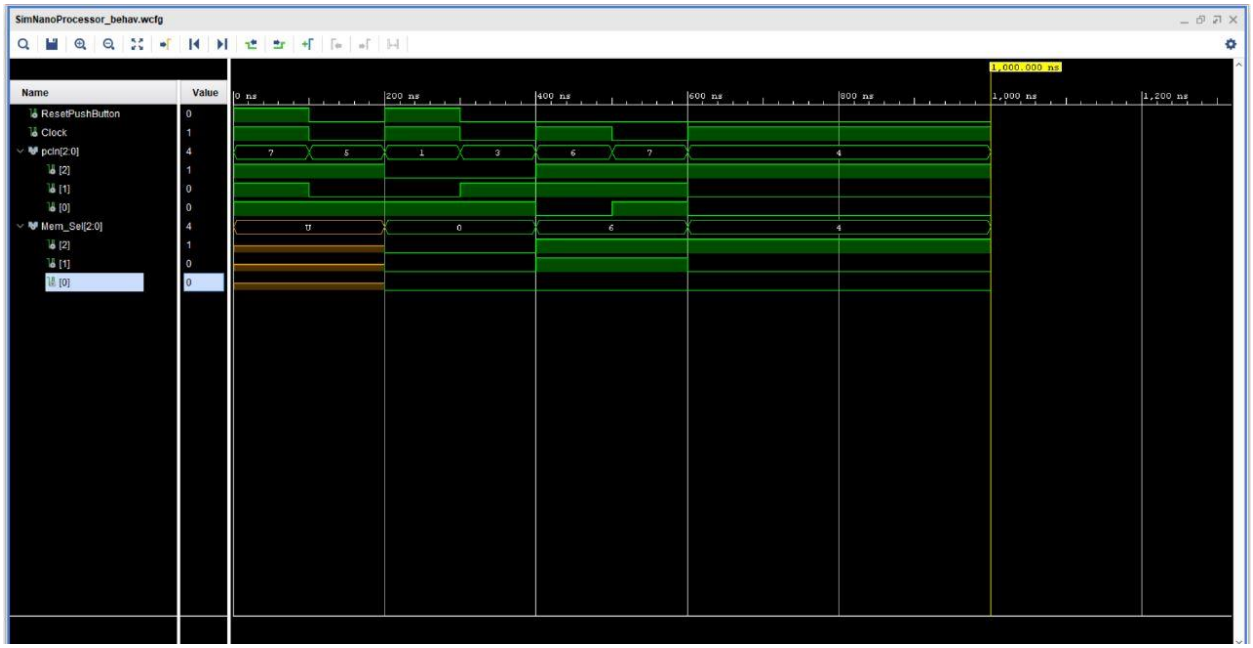
- **8 Way 4 Bit Mux**



- **Register Bank**



- **Program Counter**



5) Conclusions from the lab

In conclusion, the lab provided an opportunity to design and develop a 4-bit nanoprocessor capable of executing a set of instructions. Through collaborative teamwork, we successfully accomplished the assigned lab tasks and gained valuable insights into computer organization and digital design.

We designed and implemented essential components, such as the 4-bit add/subtract unit, 3-bit adder, 3-bit program counter, k-way b-bit multiplexers, register bank, program ROM, and instruction decoder. These components were carefully integrated to create a functional nanoprocessor.

By writing an Assembly program to calculate the total of integers between 1 and 3, we effectively tested the functionality of our nanoprocessor. We converted the Assembly program into machine code and stored it in the ROM. The execution of the program on the BASYS 3 development board demonstrated the correct functioning of our design.

Throughout the lab, we encountered challenges related to component integration, timing considerations, and ensuring the proper activation of modules based on instructions. However, by implementing efficient logic and thoroughly testing each component, we were able to overcome these challenges and achieve the desired functionality.

Overall, this lab provided a valuable hands-on experience in designing a simple nanoprocessor, enhancing our understanding of computer organization, digital design principles, and teamwork skills. The lab not only allowed us to apply theoretical concepts but also encouraged creativity and critical thinking to optimize our design and explore extra credit features.

By completing this lab, we have acquired practical skills in building and testing digital circuits, and we have gained a deeper appreciation for the complexity and intricacies of processor design.

6) Clearly describe the contribution of each team member to project and number of hours spent

- ✓ Nandaka T.H.T.D.(210406M)
    - Contribution : Worked on the design and development of the register bank and multiplexers including 8 way 4 bit ,2 way 4 bit and 2 way 3 bit .As well as participated in integrating and connecting the components inside the nanoprocessor and implementing simulation files.
    - Number of Hours spent : approximately 22 hours .

- ✓ Muthuwana M.A.N.R (210400N)

    Contribution : Focused on the design and implementation of 3 bit adder ,Instruction decoder , program counter , program ROM and Adder substracter unit .As the member who played the key role in the project ensured the way which these components should be connected.
    - Number of hours spent : approximately 28 hours.

Both team members actively collaborated, shared responsibilities, and integrated their individual contributions to create a successful final design. The total number of hours invested by the team collectively was approximately 50 hours.